(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
18 January 2001 (18.01.2001)

PCT

(10) International Publication Number
WO 01/05116 A2

(51) International Patent Classification[7]: H04L 29/00

(21) International Application Number: PCT/US00/19243

(22) International Filing Date: 13 July 2000 (13.07.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/143,457 13 July 1999 (13.07.1999) US

(71) Applicant *(for all designated States except US)*: **ALTEON WEB SYSTEMS, INC.** [US/US]; 50 Great Oaks Road, San Jose, CA 95119 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **MCKEON, James** [US/US]; 565 Hacienda Avenue #206, Campbell, CA 95008 (US). **SCHMALTZ, Dean** [US/US]; 541 Tramway Drive, Milpitas, CA 95035 (US).

(74) Agents: **GLENN, Michael, A.** et al.; Glenn Patent Group, Suite L, 3475 Edison Way, Menlo Park, CA 94025 (US).

(81) Designated States *(national)*: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *Without international search report and to be republished upon receipt of that report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 01/05116 A2

(54) Title: METHOD AND APPARATUS FOR CONDUCTING THE IP LONGEST PREFIX MATCH SEARCH

(57) Abstract: A method and apparatus for routing data packets over an IP network reduces processing overhead of making forwarding decisions by conducting the longest matching prefix search according to an optimization algorithm, the optimization algorithm being implemented on a layer three engine: a programmable hardware component. A portion of the destination address of a data packet is used to form a prefix of predetermined length. A record returned from a search of a Direct Lookup Table indicates that routes are present in a Forwarding Table that share that prefix and what their lengths are. The Forwarding Table, a translation of a conventional routing table, is traversed, starting with the longest prefix until a match is found, the longest matching prefix. A delivery Table and a Forwarding Cache contain the results of previous forwarding lookups. A packet bearing a destination address found in the Delivery Table or the Forwarding Cache is forwarded without performing a longest matching prefix search at all. Thus, the overhead of a single forwarding decision is amortized over a large number of data packets. The apparatus for routing data packets is embodied as a switch, the switch including a master processor, one or more memories, and one or more gigabit network engines, hardware components, each one including a control processor and a layer three engine.

METHOD AND APPARATUS FOR CONDUCTING THE IP LONGEST PREFIX MATCH SEARCH

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

The invention relates to routing of messages across computer networks. More particularly the invention relates to a method and apparatus for routing data packets across an IP (Internet Protocol) network in which the processing overhead of forwarding decisions is reduced through an optimized direct lookup algorithm for conducting an IP longest prefix match search.

DESCRIPTION OF THE RELATED ART

In conventional IP (Internet Protocol) networks, messages, usually called data packets, are routed from source to destination through a series of routers that receive the data packet, read the destination address, and re-transmit the data packet. If the endstation is directly connected to the router, the packet is delivered to its final destination. Otherwise, the packet is transmitted to another router that, in turn, passes the data packet along, until it reaches it's final destination. Each intermediate destination is termed a "next hop." The router determines the next hop by searching a routing table, the routing table consisting of a listing of all possible routes available to that router for forwarding the data packet. With the advent of Classless Interdomain Routing (CIDR), it became possible to determine the next hop by finding the entry in the routing table that matched the largest number of consecutive bits in a packet's destination address, known as the prefix, starting with the most significant bit of the address (bit 31).

A conventional routing table lookup is performed in a linear fashion. Each entry in a routing table is examined for a match. The operation must continue throughout the table, even after a match is found, to assure that the best match was found. Since a routing table may have hundreds of thousands of records, this process is very time-consuming. Consequently, the processing overhead imposed by routing table lookups has constituted a serious performance bottleneck in devices supporting IP routing.

Various approaches to eliminating this performance bottleneck are described in the art. For example, J. Turner, G. Varghese, M. Waldvogel, *Scalable high speed IP routing lookups,* U.S. Patent No. 6,018,524 (January 25, 2000)

describe a system in which the routing table is divided into several small databases, arranged by prefix length. A multi-step binary search algorithm sorts through the databases to determine a best matching prefix. The described method produces a marked increase in search efficiency, allowing lookups that are exponentially faster. Furthermore, H. Tzeng, *Method for IP routing table lookup*, U.S. Patent No. 6,061,712 (May 9, 2000) describes a similar method in which the routing table is organized as a series of radix trees. A portion of a data packet's IP destination address is used as an index to a RAM whose output is a pointer to the root node of one of the radix trees. A binary algorithm searches the radix tree until the best match is found. The ability to restrict the search to a single data structure, plus the speed of the binary search allows a greatly accelerated routing table lookup. A disadvantage to both the Turner and the Tzeng teachings are their multiple data structures. The creation and maintenance of these multiple data structures imposes an administrative overhead. Updating the data structures when new routes become available or removing old routes imposes a particularly heavy administrative burden. Additionally, even though indexing and subnet masking allow the search to be limited to a subset of the total available routes, the algorithms must iteratively search the subset and save all possible matches until a best match is found.

Y. Rekhter, *Efficient packet forwarding arrangement for routing packets in an internetwork*, U.S. Patent No. 5,917,820 (June 29, 1999) describes a packet forwarding arrangement utilizing a tag exchange protocol. Neighboring routers exchange unique tags, which are used to index a database of routing information. The tags are associated with predetermined groups of destination addresses in a fast-lookup tag database. The small size of the tags coupled with the table structure of the database allows lookups to be quickly and efficiently performed. A forwarding information database allows routing lookups to be performed in a conventional manner for neighboring routers that do not support the described protocol. The advantages of the Rekhter teachings, however may only be realized in small, locally-controlled networks, since the effectiveness of the protocol relies on it's being implemented in a number of neighboring routers. Since implementation of the protocol on a large scale might never happen, it would be unsuitable for use in large public networks such as the Internet. The redundant data structures required to support conventional routers and those supporting the disclosed protocol impose an undesirable administrative burden.

It would be advantageous to provide a method for conducting an IP longest prefix match search that minimizes administrative overhead by keeping creation

and maintenance of data structures to a minimum. It would be a further advantage to limit the number of data structures it is necessary to traverse during the longest prefix match search, thus greatly increasing search speed and further minimizing processing overhead. It would be desirable to provide a way of searching available routes so that the search could be terminated as soon as a match is found instead of searching all available routes to find the best match. It would be an advantage to provide a way of determining the next hop for a message without exchanging messages with other routers in a network or imposing additional protocols.

SUMMARY OF THE INVENTION

The invention provides a method and apparatus for routing data packets over an IP (Internet Protocol) network in which processing overhead of making forwarding decisions is greatly reduced. The longest matching prefix search is conducted according to an optimized algorithm incorporating direct lookup, the optimization algorithm being implemented on a Layer Three Engine, a programmable hardware component. A Direct Lookup Table and a Forwarding Table are provided. The Forwarding table is a translation of a conventional routing table constructed as a hash table with overflow. A portion of the destination address of a data packet is used to form a prefix of predetermined length. The prefix is used to address the Direct Lookup Table. A record is returned that indicates what routes are present in the Forwarding Table that share that prefix and what their lengths are. A hash is constructed from the destination address and used to address the Forwarding Table, starting with the longest prefix indicated by the record from the Direct Lookup Table. Since the search is conducted according to decreasing prefix length, the first match found is the longest matching prefix. Thus, the search is terminated after the first match is found.

A Delivery Table contains delivery information for all IP destination addresses that have required an ARP (Address Resolution Protocol) lookup; in other words, those that are on a subnet directly connected to the routing apparatus. Additionally, a Forwarding Cache containing the results of previous forwarding lookups is maintained. Before a longest matching prefix search is initiated, the Delivery Table and the Forwarding Cache are searched for records bearing a destination address identical to that of the data packet. If a matching record is found, the packet is forwarded according to instructions found in the record, eliminating the necessity of performing a longest matching prefix search at all.

Thus, the overhead of performing a single ARP lookup or a longest matching prefix search is amortized over a large number of data packets.

The apparatus for routing data packets is embodied as a switch, the switch including a master processor, one or more external memories, and one or more Gigabit Network Engines; hardware components, each one including a control processor and a Layer Three Engine.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 provides a block diagram of a switch supporting IP routing, according to the invention; and

Figure 2 provides a flowchart of a method for routing data packets across an IP network according to an optimized algorithm incorporating direct lookup, wherein the processing overhead of forwarding decisions is greatly reduced, according to the invention.

DETAILED DESCRIPTION

In the following description of the exact and longest-prefix match algorithms, several logical tables are introduced:

- Direct Lookup Table – a shorthand table to optimize the longest prefix match search.
- Forwarding Table – a translation of a conventional IP routing table providing information required to complete the longest prefix match search.
- Delivery Table – a table of information required to complete packet delivery to an endstation on a subnet directly connected to the switch.
- Forwarding Cache – the cache of previous forwarding decisions for particular IP destination addresses accessible through next hop routers.

Although these tables are logically distinct, they may be implemented in memory as one or more physical tables.

Within the art, routing tables are often interchangeably referred to as forwarding tables, or routing lookup tables or forwarding lookup tables, and so on. Within the context of the invention a "routing table" refers to a conventional IP routing table, and a "forwarding table" refers to the table described herein above. For clarity, "forwarding table" will be hereinafter capitalized and will be understood to refer to the Forwarding Table of the invention.

Referring now to Figure 1, shown is the invented apparatus for routing data packets. The invention comprises a switch 10. The switch 10 includes a master processor 11 in communication with a Gigabit Network Engine 12 (GNE). The GNE is an integrated circuit that incorporates a control processor 13 and a Layer Three engine 14 (LTE), the LTE and the control processor being in communication with each other. The LTE is the hardware implementation of the Longest Prefix match algorithm and constitutes a programmable hardware component programmed to execute the various steps of the Longest Prefix Match algorithm. The LTE has other functions, to be explained in detail in the following description. The switch 10 further includes an external memory 15. As indicated by the double pointed arrows, the master processor 11 and the GNE 12 are in communication with the external memory. For the purpose of illustration the switch is shown with only one GNE. In actual practice, however, a switch may be provided with a large number of GNE's. The switch may also have more than one external memory. The functions and interrelationship of the switch and its various components will be described in detail in conjunction with the description of the invented method.

LONGEST PREFIX MATCH WITH DIRECT LOOKUP

Referring now to Figure 2, in overview, the steps of the Longest Prefix match algorithm are:

- Receiving a data packet over an IP network (20);
- Parsing the destination address (DA) of the data packet (21);
- Searching the Direct Lookup Table (22);
- Searching the Forwarding Table (23); and

- optionally, Forwarding the data packet according to instructions contained in a record from the Forwarding Table (25).

The various steps of the algorithm are described in greater detail below.

The goal of the invented method is to reduce the complexity of the longest prefix search while preserving acceptable performance. The workings of IP networks are well known in the art. The steps of receiving the data packet 20 and parsing the destination address 21 will be apparent to those skilled in the art. The upper portion of the IP address, constituting the first N bits, is used to address a data structure 22, the Direct Lookup Table. The search returns a record indicating what routes are to be found within the Forwarding Table that share the prefix formed from the DA and what their lengths are. The portion of the upper part of the DA used as an index is programmable, and may vary from zero to 16 bits in the current embodiment, resulting in tables having from 1-64K records. However, the actual length of the portion of the DA addressed used as an index may vary over a greater range according to the actual bit length of the DA. The limit of 16 bits is used in the current embodiment because it subsumes 16-bit prefixes, which statistics indicate is a common prefix length, and because 64K records constitutes the largest practical size for direct addressing.

The upper N bits of the IP DA are used to directly address (22) a 2^N table of 4-byte records, the Direct Lookup Table. The records of the Direct Lookup Table, termed forwarding records, are of the form:

L-1		N (N – 1)	0
Prefix length map			Longest Matching Length

L = the bit length of the DA.

[(L-1):N] Prefix Length Map: Each bit that is set represents a prefix length that appears in the Forwarding Table and that shares the upper N bits with the DA. It provides a key to search for longer prefixes. For example, where L=32, if bit 31 (L-1) is set, it represents a 32-bit prefix in the Forwarding Table. If bit 30 is set, that represents a 31-bit prefix and so on. If no bits are set in the Prefix length map, it indicates that no longer prefixes exist in the Forwarding Table.

[(N–1):0] Longest Matching Length: The longest matching prefix length of the forwarding table record shorter than or equal to N. Because the first N bits of the prefix form an unambiguous address into a portion of the routing table, the longest matching prefix shorter than or equal to N can be determined during the construction of this table. The length of this prefix is stored into the Longest Matching Length field.

Shown below is an exemplary forwarding record. As previously indicated, in the preferred embodiment of the invention, N = 16, therefore bits 31 – 16 of the record below constitute the Prefix Length Map.

Record = 0000_0000_0000_0110_1000_0000_0000_0000.

It is apparent that bits 18 and 17 are set, indicating that there are prefixes nineteen and eighteen bits in length in the Forwarding Table that share the upper sixteen bits of the destination address.

Bits 15 – 0 constitute the Longest Matching Length. Bit 15 is set, therefore the longest matching length to the upper 16 bits of the DA is sixteen bits.

The search procedure is the same regardless of how many of the address bits are used for direct addressing.

SEARCH PROCEDURE

- The upper N bits of the DA are used to address (22) the Direct Lookup table of 2^N records directly, and a forwarding record is returned.

- The information from the returned forwarding record is used to retrieve (23) one record from the Forwarding Table that will contain the necessary information to complete the route. Referring to Figure 1, the bi-directional arrow indicates that the route information of the Forwarding Table originates from a conventional routing table (24).

- The logic processes the packet (25) according to the instructions listed in the forwarding table record.

If the longest matching prefix is N bits or less, no bits in the Prefix Length Map are set. The Longest Matching Length is then used to retrieve the Forwarding Table record. For prefixes that do not have a longest match in the Forwarding Table, the memory pointer is encoded to indicate a default route.

If prefixes exist that are longer than N bits, bits are set in the Prefix Length Map. In this case, the Prefix Length Map is employed to search the Forwarding Table. Each bit that is set in the Prefix Length Map represents a prefix length, from (N + 1) to 32 bits. The Longest Matching Length is retained in memory in case the given DA does not match records at any of these lengths.

The hardware begins searching the forwarding table for longer matching prefixes, starting with the longest indicated prefix. It uses the significant bits of the DA (using the prefix length specified by the Prefix Length Map as a mask) to construct a hash and address the Forwarding Table. The Forwarding Table is constructed as a hash table with overflow. The linked list beginning with each hash entry is traversed until either a match with the DA is found or all the overflow records are exhausted. A match indicates the end of the search. Exhausting the search causes the hardware to return to the Prefix Length Map and repeat the search with the next longest prefix length from the Prefix Length Map.

If the lengths indicated by the Prefix Length Map are searched without obtaining a match, the Longest Matching Length field is used to determine the next search length, and the route is then obtained. In the special case that no bits are set in the Longest Matching Length field the least significant four bits of the DA (bits 3-0) are used to select among sixteen default routes.

To create and maintain the Forwarding Table and Direct Lookup Table, software parses the routing table that is received by the switch as part of the chosen routing protocol. Based upon the prefix length chosen to index the Direct Lookup Table, the software determines the entire set of routes that share this prefix and collects their lengths in the Prefix Length Map. The Forwarding Table is created by creating a hash on the represented prefixes and adding a linked list of overflow records when hashes are identical. These two tables are then loaded into a section of memory and the hardware searches them as described above.

NUMERICAL EXAMPLES

An example topology and routing situation is described below to illustrate the forwarding algorithm. Here, a Direct Lookup Table indexed by the upper 16 bits of the DA is assumed.

Networks (binary notation): A = 10000000.00000001/16
B = 10000000.00000001.01/18
C = 10000000.0000000l.11/18
D = 10000000.00000010.000/19
E = 10000000.00000001.111/19

Endstations:
A1 =.10000000.00000001.00000000.00000011
B1 = 10000000.00000001.01000000.00000100
C1 = 10000000.00000001.11000000.00000001
D1 = 10000000.00000010.00000000.00001000
X1 = 10000001.00000000.00000000.00000001
A2 = 10000000.00000001.10000000.00000010
E1 = 10000000.00000001.11100000.00000011

Example 1: IPDA=A1

After missing the lookup in the Forwarding Cache, the search of the Direct Lookup table returns the following record:

Record = 0000_0000_0000_0110_1000_0000_0000_0000.

This record indicates that there are longer matching prefixes possible, and also returns the length of A, which is the best match in case searching for longer prefixes fails. The longest prefix possible is 19 bits, so a hash is constructed with the upper 19 bits of the DA and that portion of the forwarding table is searched. This search will fail to obtain a match, so a search of the 18 bit prefixes is attempted. This also fails. There are no indicated prefixes at the 17-bit length, therefore the best match is at length 16. The length 16 is used to compute the hash and retrieve the correct forwarding record. The minimum number of memory references to complete the forwarding decision is 5.

Example 2: IPDA=B1

The forwarding record is the same as that retrieved for Example 1, since the upper 16 bits of B1 is identical to A1. The prefix search begins with a length of 19, misses, and searches again at length 18. This time the record hits in the table,

so the best prefix is now found. The minimum number of references to complete the forwarding decision is 4.

Example 3:IPDA = D1

The forwarding record is:

Record = 0000_0000_0000_0100_0000_0000_0000_0000.

Since there is the possible match at D, the 19-bit prefix length search is done first. In this case, a match exists. Because there were no routes shorter than 19, however, no other lengths are indicated. If the match had not existed, the default route would have been chosen based on the lower four bits of the IP destination address.

Example 4: IPDA=X1

The forwarding record is:

Record = 0000_0000_0000_0000_0000_0000_0000_0000.

All-zeroes within the Prefix Length Map indicates that no routes longer than 16 bits exist in the table. The all-zeroes value for the Longest Matching Length indicates that no routes shorter than 16 bits exist with this prefix either, so, again, the default route would have been chosen.

Example 5: IPDA=A2

The forwarding record is identical to that of the first example. The search is conducted beginning with a prefix length of 19. Since there is no match with networks B, C, D, or E, however, the Longest Matching Length is used. In this case, the 16-bit prefix is the best match.

Example 6: IP DA = E1

The forwarding record is identical to that of the first example. The search begins with a prefix length of 19, and a match is obtained for that prefix. The search is now complete. Although there were other matches at lengths of 16 and 18 (networks A and C), network E is the longest, and therefore, the best match.

While the invention has been described herein in relation to 32-bit IP addresses, such description is exemplary only, and not intended to be limiting. Those skilled

in the art will appreciate that the invention may easily be implemented in network environments other than an IP network. Furthermore, the invention may be applied to destination addresses of any bit length.

DELIVERY TABLE

In the event that the forwarding table record indicates that the DA is on a subnet directly connected to a switch, the logic delivers the packet, or frame to the Control Processor to initiate an ARP (Address Resolution Protocol). In order to maintain maximal performance, it is necessary to reduce the average ARP overhead for traffic to directly connected endstations as much as possible. As indicated in Figure 1, after the destination address is parsed 11, an exact search algorithm searches a Delivery Table and a Forwarding Cache 26 to see if an exact match to the DA may be found among their records. In the event that a match is found, the packet is forwarded 25 according to the instructions found in the record, and the necessity of performing a Longest Matching Prefix search is averted. If the search fails, the Longest Matching Prefix algorithm is invoked.

The Delivery Table, constructed as a hash table with overflow, contains delivery information for all IP destination addresses that have required an ARP lookup to the Master Processor. Its purpose is to amortize the overhead of conducting an ARP request over a large number of packets, which appreciably reduces the average overhead of an ARP request to a directly connected endstation. A record is added to the Delivery Table for every packet that requires an ARP lookup to the Master Processor. The structure of a Delivery Table record includes information required to identify the record as an exact match for a given IP address, as well as control information for determining the frame's subsequent delivery and/or processing.

The Delivery Table is traversed as follows:

- A hash is constructed based upon the entire 32-bits of the IP DA of the packet in question. This hash is added to a base index to form a physical memory address in the Delivery Table.

- The corresponding Delivery Table record is retrieved. Various outcomes are possible:

1. The table record matches the IP DA. The packet is delivered according to the information in the table record.

2. The table record does not match the IP DA and there are no overflow records, or there is no table record at that hash address. The search has failed, so the longest prefix match process is initiated.

3. The table record does not match the IP DA, and there are overflow records. The overflow records are searched linearly until either 1 or 2 are valid.

As previously indicated, if the Delivery Table search fails, the longest prefix match search is invoked. If the subnet that this search identifies as the longest match is directly connected to the switch, the GNE Control Processor needs to complete the delivery process by initiating an ARP request to the Master Processor. The LTE delivers the frame to the Control Processor with instructions to do this. The Master Processor maintains a global ARP cache created through the requests of all the GNE's within the switch. If the given DA does not exist in this table, the Master Processor issues an ARP request to the port(s) connected to that subnet, receives the result, and then passes it back to the requesting GNE.

If subsequent frames with an identical DA arrive at the ingress GNE prior to the Control Processor receiving the ARP information back from the Master Processor, they are also delivered to the Control Processor for ARP resolution. To prevent the Control Processor from issuing multiple requests to the Master Processor for ARP information on the same DA, a dummy record is added to the Delivery Table after the first frame is delivered to the Control Processor. This dummy record also causes delivery of subsequent frames with the identical DA to the Control Processor, but it contains a special field that informs the Control Processor that an ARP request is pending.

After the Control Processor receives the ARP information from the Master Processor, it must update the local Delivery Table. It does this by writing the record information into a memory within the LTE and then enabling the LTE to enter the record into the table. If a record already exists at that slot in the table, the LTE finds the end of the linked list of overflow records, acquires a new buffer from the entity managing the overflow, and adds the record to the list. In principle, the Delivery Table should have an unlimited degree of overflow, since the penalty for missing in the table and conducting an ARP lookup is high. In practice,

however, a limit on the length of the table search is instituted to prevent endless loops caused by bad pointer construction, which in turn limits the degree of overflow.

An overflow manager controls the buffer space available for overflow records to the Delivery Table. This function is designed into the frame buffer manager and interfaces with the LTE.

The Master Processor monitors endstation moves that require modifications to the Delivery Table and communicates such modifications to each GNE. Details of the replacement policy appear below.

FORWARDING CACHE

The longest prefix match search requires multiple memory references to discover a match in most cases. A means of amortizing the overhead of the longest prefix match is desirable. A common technique to achieve this is to cache the results of previous forwarding lookups and apply those results to subsequent packets with identical destinations. The Forwarding Cache refers to a table of such stored lookups. The term cache here refers not to an on-chip memory, but to a table of route results for particular 32-bit IP addresses that may be searched with an average close to a single memory reference.

The construction of the Forwarding Cache is very similar to that of the Delivery Table. Both are indexed based upon a hash of the 32-bit DA, and both must be searched in the course of completing a forwarding decision. The Forwarding Cache record is of the same form as a Delivery Table record. The search procedure for finding a record in the Forwarding Cache is also identical to that of the Delivery Table. Given these similarities, the Forwarding Cache and the Delivery Table are collapsed into a single physical table. The major benefit is a reduction in the number of memory references required to complete processing on the frame.

Following the completion of a longest prefix match search, the hardware assembles the fields of the Forwarding Cache record and writes them into an available slot within the cache.

The cache size is chosen based upon the traffic pattern assumptions. The life span of a Forwarding Cache record is not identical to that of a Delivery Table record. Contrasted with endstation locations, forwarding paths (routes) change much more dynamically. Potentially, every update of the routing protocol contains some change in the routing table, although the Master Processor delays updating the forwarding tables until a certain interval has passed between updates or some quota of routes has changed.

A programmable degree of overflow is provided for Forwarding Cache records, preferably 0-3 levels, since long overflow chains reduce the effectiveness of the cache.

MAINTENANCE OF FORWARDING CACHE AND DELIVERY TABLE

The difference in the maintenance characteristics between the Delivery Table and the Forwarding Cache complicates the task of table maintenance when they are merged into a single physical table. It is necessary to distinguish between the two types of table record when completing the replacement procedure. For the purposes of this discussion, the term maintenance refers to table operations other than additions, such as the selective removal of records or the wholesale flushing of the table.

The expected number of directly connected endstations can be stored in their entirety within the Delivery Table, and there is no protocol specification that demands the records age out after a particular time. These records should be replaced or updated only when their information becomes stale, which occurs only after certain routing table updates, or changes in the Master Processor's ARP cache, or L2 spanning tree reconfigurations. Following these types of events, the Delivery Table is either erased or selectively updated, depending upon details of the software implementation. A mechanism allows the Master Processor to maintain the state of its ARP cache and to communicate all changes to the local Delivery Tables. Otherwise, the local records remain static within the tables, and are not replaced when their hash addresses collide with Forwarding Cache records or other Delivery Table records.

In contrast, because the potential number of IP DA's seen by the switch is far larger than the size of the Forwarding Cache, Forwarding Cache records must be regularly replaced. However, there is no global timeout requirement within the

routing protocols that would require cache records to be invalidated after a particular period, so it is not necessary to preemptively age the cache. Additionally, several connections between the same endstations may occur over a longer period, so that the value of the cache record is extended over this entire period. Therefore, a simple replacement policy is implemented that allows cache records to overwrite one another when they address the same cache slot. Most records will exist long enough to be hit during some number of TCP connections, and they will be replaced eventually by a more current record.

The mixed table is designed to always prefer Delivery Table records to Forwarding Cache records. If a Delivery Table record is to be added to the table, and it collides with a Forwarding Cache record, the Forwarding Cache record is overwritten.

The hardware conducts all updates of this mixed table. Updates are the addition of a new element to the table or deletion or modification of an existing element. Because the table is implemented as a hash table with overflow, updates are not necessarily performed in a single memory reference. For the addition of a Delivery Table record, the hardware must check the hash location for Delivery Table records already in existence. Once it discovers the last element on the linked list, it requests a free overflow buffer from the overflow manager, updates the overflow pointer of the old last element, and stores the new record in the free overflow buffer. For deletion of a Delivery Table record, the hardware must find the record, write its overflow pointer into the overflow pointer field of the previous record in the list, and then pass the overflow buffer back to the overflow manager.

Forwarding cache updates are performed similarly. The Forwarding Cache update logic has multiple cache slots (hash location plus the programmable number of overflow locations) to choose from when performing replacements. In these cases, the logic uses an incrementing counter to select which slot to replace. This policy reduces the incidence of thrashing between two active DA's that hash to the same address. During the search of the Forwarding Cache/Delivery Table prior to the longest-prefix match search, the hardware saves whether the searched locations are part of the Delivery Table or the Forwarding Cache. For an addition to the cache, the hardware uses this information to guide the replacement, replacing the location specified by the counter if possible, or if this location is occupied by a Delivery Table record, replacing the next available sequential location may not be possible to make an addition if the number of

Delivery Table records at that hash slot is larger than the depth of the Forwarding Cache. Deletions are performed identically to those of Delivery Table records.

The hardware provides a unified structure that allows the Control Processor to add or delete records from any of the hash tables. On an addition, the Control Processor writes the entire 16-byte table record into a reserved memory, signals which table it wishes to add to, and enables the hardware to perform the operation. On a deletion, the Control Processor specifies at least the prefix and length of the record so that the hardware may construct the hash, identify the table, and enable the hardware. Modifications to existing records are also hardware-assisted in a similar manner. When the operation is complete, the hardware sets an event to signal the completion of an addition, deletion, or modification operation.

Although the invention is described herein with reference to certain preferred embodiments, one skilled in the art will readily appreciate that other applications may be substituted for those set forth herein without departing from the spirit and scope of the present invention. For example, the invention may be used with destination addresses of any bit length, such as the new 128-bit IP addresses. Accordingly, the invention should only be limited by the Claims included below.

CLAIMS

What is claimed is:

1. A method of conducting an IP longest prefix search of an IP (Internet Protocol) routing table, comprising the steps of:

receiving a data packet at a switch, said data packet bearing a destination address;

parsing said destination address

providing a forwarding table, said forwarding table comprising a translation of said IP routing table;

providing a direct lookup table, said direct lookup table comprising a plurality of forwarding records;

using a prefix of a predetermined number of upper bits taken from said destination address to address said direct lookup table, and returning a forwarding record, wherein said forwarding record indicates what routes are present in said routing table that have a prefix identical to said portion of said destination address, and what their lengths are;

searching said forwarding table for records bearing longer matching prefixes, said forwarding table comprising a hash table with overflow; and

when a match is found, processing said data packet according to instructions listed in said forwarding table record.

2. The method of Claim 1, further comprising the optional step of:

initiating an ARP (Address Resolution Protocol) lookup when a destination address represents an endstation on a subnet directly connected to said routing apparatus.

3. The method of Claim 2, wherein said predetermined number = N, and wherein the bit length of said destination address = L.

4. The method of Claim 3, wherein said forwarding records are L bits in length, and wherein the upper (L – N) bits constitute a prefix length map, and a bit that is set from among the remaining bits indicates a longest matching length.

5. The method of Claim 4, wherein each bit in said prefix length map that is set represents a prefix length that appears in said routing table and that shares the upper N bits with said destination address.

6. The method of Claim 6, wherein said longest matching length constitutes a longest matching prefix to said prefix of N bits from said destination address.

7. The method of Claim 6, wherein said longest matching length constitutes a longest matching prefix to said prefix of N bits from said destination address.

8. The method of Claim 7, wherein said forwarding table search step comprises the steps of:

parsing said prefix length map to determine the prefix lengths present in said routing table;

choosing the longest prefix length available;

constructing a hash from the destination address and said longest prefix length;

addressing said forwarding table and optionally, an overflow, using said hash;

if a match is found, terminating said search and proceeding to said processing step.

9. The method of Claim 8, wherein said forwarding table search step further comprises one or more of the steps of:

if a match is not found, returning to said prefix length map and using the next longest prefix length available to search;

repeating said search step with progressively shorter prefix lengths until a match is found;

if a match is not found, using said longest matching length to determine a next search length; and

if no bits are set for a longest matching length, using the least significant bits of said destination address to select from a plurality of default routes.

10. The method of Claim 8, wherein said forwarding table search step comprises the steps of:

if no bits are set in said prefix length map, determining said longest matching length;

constructing a hash from said destination address and said longest matching length;

addressing said forwarding table and optionally, an overflow, using said hash;

if a match is found, terminating said search and proceeding to said processing step.

11. The method of Claim 10, wherein said forwarding table search step further comprises the step of:

if no bits are set for a longest matching length, using the least significant bits of said destination address to select from a plurality of default routes.

12. The method of Claim 2, further comprising the preliminary steps of:

providing a Delivery Table, said Delivery Table comprising delivery information for destination addresses that have required one of said ARP lookups;

providing a Forwarding Cache, said Forwarding Cache comprising a table of results of previous longest prefix match searches; and

searching said Delivery Table and said Forwarding Cache for records with an IP destination address that matches said destination address of said received data packet

13. The method of Claim 12, wherein said Delivery Table and said Forwarding Cache together comprise a single physical table, said single physical table comprising Delivery Table records and Forwarding Cache records, said table being a hash table with an overflow, said records being indexed based upon a hash of the entire destination address in each of said records.

14. The method of Claim 13, wherein said Delivery Table and Forwarding Cache-searching step comprises the steps of:

constructing a hash based upon the entire destination address of said data packet and adding said hash to a base index to form a physical memory address of a record in said single table; and

retrieving a table record at said physical memory address.

15. The method of Claim 14, further comprising one of the steps of:

if said table record matches said destination address of said data packet, delivering said packet according to information in said table record;

if said table record doesn't match said destination address of said data packet, and there are no matching overflow records, initiating a longest prefix match;

if no table record at said address exists, initiating a longest prefix match; and

if the table record doesn't match said destination address of said data packet, searching said overflow records linearly until at least one valid match is found, and delivering said packet according to information in said overflow record.

16. A method of routing data packets over an IP (Internet Protocol) network, wherein processing overhead of forwarding decisions is reduced, comprising the steps of:

receiving a data packet at a routing apparatus, said data packet bearing a destination address;

parsing said destination address;

conducting a longest prefix match search according to an optimized algorithm, said algorithm incorporating direct lookup, said longest prefix comprising a route in an IP routing table that matches the largest number of consecutive bits in said destination address;

forwarding said data packet according to said longest prefix.

17. The method of Claim 16, further comprising the optional step of:

initiating an ARP (Address Resolution Protocol) lookup when a destination address represents an endstation on a subnet directly connected to said routing apparatus.

18. The method of Claim 17, wherein said optimized algorithm comprises the steps of:

providing a direct lookup table comprising a plurality of forwarding records;

providing a forwarding table, said forwarding table a translation of said routing table;

using a prefix of a predetermined number of upper bits taken from said destination address to address said direct lookup table, wherein said lookup table is indexed by prefixes of the same length, and returning a forwarding record, wherein said forwarding record indicates what routes are present in said routing table that have a prefix identical to said portion of said destination address, and what their lengths are;

searching said forwarding table for records bearing longer matching prefixes, said forwarding table comprising a hash table with overflow; and

when a match is found, processing said data packet according to instructions listed in said forwarding table record.

19. The method of Claim 18, wherein said predetermined number = N, and wherein the bit length of said destination address = L.

20. The method of Claim 19, wherein said forwarding records are L bits in length, and wherein the upper (L– N) bits constitute a prefix length map, and a bit that is set from among the remaining bits indicates a longest matching length.

21. The method of Claim 20, wherein each bit in said prefix length map that is set represents a prefix length that appears in said routing table and that shares the upper N bits with said destination address.

22. The method of Claim 21, wherein said longest matching length constitutes a longest matching prefix to said prefix of N bits from said destination address.

23. The method of Claim 22, wherein said forwarding table search step comprises the steps of:

parsing said prefix length map to determine the prefix lengths present in said routing table;

choosing the longest prefix length available;

constructing a hash from the destination address and said longest prefix length;

addressing said forwarding table and optionally, said overflow, using said hash;

if a match is found, terminating said search and proceeding to said processing step.

24. The method of Claim 22, wherein said forwarding table search step further comprises one or more of the steps of:

if a match is not found, returning to said prefix length map and using the next longest prefix length available to search;

repeating said search step with progressively shorter prefix lengths until a match is found;

if a match is not found, using said longest matching length to determine a next search length; and

if no bits are set for a longest matching length, using the least significant bits of said destination address to select from a plurality of default routes.

25. The method of Claim 22, wherein said forwarding table search step comprises the steps of:

if no bits are set in said prefix length map, determining said longest matching length;

constructing a hash from said destination address and said longest matching length;

addressing said forwarding table and optionally, said overflow, using said hash;

if a match is found, terminating said search and proceeding to said processing step.

26. The method of Claim 25, wherein said forwarding table search step further comprises the step of:

if no bits are set for a longest matching length, using the least significant bits of said destination address to select from a plurality of default routes.

27. The method of Claim 17, further comprising the preliminary steps of:

providing a Delivery Table, said Delivery Table comprising delivery information for destination addresses that have required one of said ARP lookups;

providing a Forwarding Cache, said Forwarding Cache comprising a table of results of previous longest prefix match searches; and

searching said Delivery Table and said Forwarding Cache for records with an IP destination address that matches said destination address of said received data packet

28. The method of Claim 27, wherein said Delivery Table and said Forwarding Cache together comprise a single physical table, said single physical table comprising Delivery Table records and Forwarding Cache records, said table comprising a hash table with overflow, said records being indexed based upon a hash of the entire destination address in each of said records.

29. The method of Claim 28, wherein said Delivery Table and Forwarding Cache-searching step comprises the steps of:

constructing a hash based upon the entire IP destination address of said data packet and adding said hash to a base index to form a physical memory address of a record in said single table; and

retrieving a table record at said physical memory address.

30. The method of Claim 29, further comprising one of the steps of:

if said table record matches said destination address of said data packet, delivering said packet according to information in said table record;

if said table record doesn't match said destination address of said data packet, and there are no matching overflow records, initiating a longest prefix match;

if no table record at said address exists, initiating a longest prefix match; and

if the table record doesn't match said destination address of said data packet, searching said overflow records linearly until at least one valid match is found, and delivering said packet according to information in said overflow record.

31. The method of Claim 28, wherein a Delivery Table record is created for every received data packet requiring an ARP lookup.

32. The method of Claim 28, wherein a Forwarding Cache record is added to said single table following completion of each longest prefix match search.

33. The method of Claim 28, wherein a degree of overflow provided for said Forwarding Cache records is programmable.

34. The method of Claim 33, wherein Delivery Table records are distinguishable from Forwarding Cache records.

35. The method of Claim 34, wherein Delivery Table records may be updated following any of:

routing table updates;

changes to a master processor's ARP cache; and

L2 spanning tree reconfigurations.

36. The method of Claim 35, wherein said master processor maintains the state of said ARP cache, and communicates changes to said Delivery Table.

37. The method of Claim 34, wherein a Forwarding Cache record overwrites a previous Forwarding Cache record when both records address the same cache slot.

38. The method of Claim 34, wherein Delivery Table records take precedence over Forwarding Cache records, so that when a Delivery Table record collides with a Forwarding Cache record, the Delivery Table record overwrites that Forwarding Cache record.

39. The method of Claim 35, further comprising the step of adding a Delivery Table record to said table, said step of adding a Delivery Table record comprising the steps of:

checking a hash location for Delivery Table records already in existence;

requesting a free overflow buffer from an overflow manager after finding a last item in a linked list, said linked list comprising said overflow, for said hash location;

updating an overflow pointer for said last item; and

writing said Delivery Table record to said free overflow buffer.

40. The method of Claim 34, further comprising the step of deleting a Delivery Table record from said table, said step of deleting a Delivery Table record comprising the steps of:

finding said record to be deleted;

writing said record's overflow pointer to a record immediately preceding it in a linked list, said linked list comprising said overflow; and

passing a resulting free overflow buffer back to an overflow manager.

41. The method of Claim 34, further comprising the step of adding a Forwarding Cache record to said table, said step of adding a Forwarding Cache record comprising the steps of :

using an incrementing counter to select a slot to replace, wherein multiple cache slots are available for performing updates, said multiple slots including said hash location and said programmable number of overflow locations;

replacing said slot if said slot is occupied by a Forwarding Cache record; and

replacing a subsequent available location if said previous slot is occupied by a Delivery Table record.

42. The method of Claim 34, further comprising the step of deleting a Forwarding Cache record from said table, said step of deleting a Forwarding Cache record comprising the steps of:

finding said record to be deleted;

writing said record's overflow pointer to a record immediately preceding it in a linked list, said linked list comprising said overflow; and

passing a resulting free overflow buffer back to an overflow manager.

43. The method of Claim 18, wherein said ARP lookup initiation step comprises the steps of:

delivering said data packet, said data packet comprising a frame, to a control processor;

directing a request for said ARP lookup from said control processor to a master processor; and

searching a global ARP cache for a match with said destination address, said ARP cache comprising a table maintained by said master processor of ARP requests initiated within said routing apparatus, said routing apparatus comprising a switch.

44. The method of Claim 43, said ARP lookup initiation step further comprising one of the steps of:

if a match is found, returning the results of said cache search to said control processor;

if no match is found, issuing an ARP request by said master processor to ports connected to said subnet, receiving results of said request, and returning said results to said requesting control processor.

45. The method of Claim 44, said ARP lookup initiation step further comprising the step of:

when a subsequent frame is received at said switch having an identical destination address to a first frame for which an ARP request is pending, creating a table record for said destination address in said ARP cache indicating that a request for said address is pending, so that said control processor may receive said subsequent frame without directing multiple requests to said master processor for the same destination address.

46. A routing apparatus for routing data packets over an IP network comprising:

a master processor;

one or more network engines, each of said network engines including a control processor and a layer three engine in communication with said control processor, each of said control processors in communication with said master processor;

at least one memory accessible to said master processor, said control processors and said layer three engines;

wherein said routing apparatus is programmed to execute a method of routing data packets over an IP network, wherein processing overhead of forwarding decisions is reduced by conducting an IP longest matching prefix search according to an optimized algorithm incorporating direct lookup.

47. The routing apparatus of Claim 46, wherein said routing apparatus comprises a switch,

48. The routing apparatus of Claim 46, wherein said method of routing data packets over an IP (Internet Protocol) network comprises the steps of:

receiving a data packet at said routing apparatus, said data packet bearing a destination address;

parsing said destination address;

conducting a longest prefix match search according to an optimized algorithm incorporating direct lookup, said longest prefix comprising a route in an IP routing table that matches the largest number of consecutive bits in said destination address, wherein said layer three engine is programmed to execute said optimized algorithm; and

forwarding said data packet according to said longest prefix.

49. The routing apparatus of Claim 48, wherein said method further comprises the optional step of:

initiating an ARP (Address Resolution Protocol) lookup when a destination address represents an endstation on a subnet directly connected to said routing apparatus.

50. The routing apparatus of Claim 48, wherein said optimized algorithm comprises the steps of:

providing a direct lookup table comprising a plurality of forwarding records;

providing a forwarding table, said forwarding table a translation of said routing table;

using a prefix of a predetermined number of upper bits taken from said destination address to address said direct lookup table, and returning a forwarding record, wherein said forwarding record indicates what routes are present in said routing table that have a prefix identical to said portion of said destination address, and what their lengths are;

searching said forwarding table for records bearing longer matching prefixes, said forwarding table comprising a hash table with overflow; and

when a match is found, processing said data packet according to instructions listed in said forwarding table record.

51. The routing apparatus of Claim 50, wherein said predetermined number = N, and wherein the bit length of said destination address = L.

52. The routing apparatus of Claim 51, wherein said forwarding records are L bits in length, and wherein the upper (L – N) bits constitute a prefix length map and a bit that is set from among the remaining bits indicates a longest matching length.

53. The routing apparatus of Claim 52, wherein each bit in said prefix length map that is set represents a prefix length that appears in said routing table and that shares the upper N bits with said destination address.

54. The routing apparatus of Claim 53, wherein said longest matching length constitutes a longest matching prefix to said prefix of N bits from said destination address.

55. The routing apparatus of Claim 54, wherein said forwarding table search step of said method comprises the steps of:

parsing said prefix length map to determine the prefix lengths present in said routing table;

choosing the longest prefix length available;

constructing a hash from the destination address and said longest prefix length;

addressing said forwarding table and optionally, said overflow, using said hash;

if a match is found, terminating said search and proceeding to said processing step.

56. The routing apparatus of Claim 55, wherein said forwarding table search step of said method further comprises one or more of the steps of:

if a match is not found, returning to said prefix length map and using the next longest prefix length available to search;

repeating said search step with progressively shorter prefix lengths until a match is found;

if a match is not found, using said longest matching length to determine a next search length; and

if no bits are set for a longest matching length, using the least significant bits of said destination address to select from a plurality of default routes.

57. The routing apparatus of Claim 54, wherein said forwarding table search step of said method comprises the steps of:

if no bits are set in said prefix length map, determining said longest matching length;

constructing a hash from said destination address and said longest matching length;

addressing said forwarding table and optionally, said overflow, using said hash;

if a match is found, terminating said search and proceeding to said processing step.

58. The routing apparatus of Claim 57, wherein said forwarding table search step of said method further comprises the step of:

if no bits are set for a longest matching length, using the least significant bits of said destination address to select from a plurality of default routes.

59. The routing apparatus of Claim 49, wherein said method further comprises the preliminary steps of:

providing a Delivery Table, said Delivery Table comprising delivery information for destination addresses that have required one of said ARP lookups;

providing a Forwarding Cache, said Forwarding Cache comprising a table of results of previous longest prefix match searches; and

searching said Delivery Table and said Forwarding Cache for records with an IP destination address that matches said destination address of said received data packet

60. The routing apparatus of Claim 59, wherein said Delivery Table and said Forwarding Cache together comprise a single physical table, said single physical table comprising Delivery Table records and Forwarding Cache records, said table being a hash table with an overflow, said records being indexed based upon a hash of a 32 bit destination address in each of said records.

61. The routing apparatus of Claim 60, wherein said Delivery Table and Forwarding Cache-searching step of said method comprises the steps of:

constructing a hash based upon the entire IP destination address of said data packet and adding said hash to a base index to form a physical memory address of a record in said single table; and

retrieving a table record at said physical memory address.

62. The routing apparatus of Claim 61, said method further comprising one of the steps of:

if said table record matches said destination address of said data packet, delivering said packet according to information in said table record;

if said table record doesn't match said destination address of said data packet, and there are no matching overflow records, initiating a longest prefix match;

if no table record at said address exists, initiating a longest prefix match; and

if the table record doesn't match said destination address of said data packet, searching said overflow records linearly until at least one valid match is found, and delivering said packet according to information in said overflow record.

63. The routing apparatus of Claim 60, wherein a Delivery Table record is created for every received data packet requiring an ARP lookup.

64. The routing apparatus of Claim 60, wherein a Forwarding Cache record is added to said single table following completion of each longest prefix match search.

65. The routing apparatus of Claim 60, wherein a degree of overflow provided for said Forwarding Cache records is programmable.

66. The routing apparatus of Claim 64, wherein Delivery Table records are distinguishable from Forwarding Cache records.

67. The routing apparatus of Claim 66, wherein Delivery Table records may be updated following any of:

routing table updates;

changes to said master processor's ARP cache; and

L2 spanning tree reconfigurations.

68. The routing apparatus of Claim 64, wherein said master processor maintains the state of said ARP cache, and communicates changes to said Delivery Table.

69. The routing apparatus of Claim 64, wherein a Forwarding Cache record overwrites a previous Forwarding Cache record when both records address the same cache slot.

70. The routing apparatus of Claim 64, wherein Delivery Table records take precedence over Forwarding Cache records, so that when a Delivery Table record collides with a Forwarding Cache record, the Delivery Table record overwrites that Forwarding Cache record.

71. The routing apparatus of Claim 64, said method further comprising the step of adding a Delivery Table record to said table, said step of adding a Delivery Table record comprising the steps of:

checking a hash location for Delivery Table records already in existence;

requesting a free overflow buffer from an overflow manager after finding a last item in a linked list, said linked list comprising said overflow, for said hash location;

updating an overflow pointer for said last item; and

writing said Delivery Table record to said free overflow buffer.

72. The routing apparatus of Claim 64, said method further comprising the step of deleting a Delivery Table record from said table, said step of deleting a Delivery Table record comprising the steps of:

finding said record to be deleted;

writing said record's overflow pointer to a record immediately preceding it in a linked list, said linked list comprising said overflow; and

passing a resulting free overflow buffer back to an overflow manager.

73. The routing apparatus of Claim 64, said method further comprising the step of adding a Forwarding Cache record to said table, said step of adding a Forwarding Cache record comprising the steps of:

using an incrementing counter to select a slot to replace, wherein multiple cache slots are available for performing updates, said multiple slots including said hash location and said programmable number of overflow locations;

replacing said slot if said slot is occupied by a Forwarding Cache record; and

replacing a subsequently available location if said slot is occupied by a Delivery Table record.

74. The routing apparatus of Claim 64, said method further comprising the step of deleting a Forwarding Cache record from said table, said step of deleting a Forwarding Cache record comprising the steps of:

finding said record to be deleted;

writing said record's overflow pointer to a record immediately preceding it in a linked list, said linked list comprising said overflow; and

passing a resulting free overflow buffer back to an overflow manager.

75. The routing apparatus of Claim 50, wherein said ARP lookup initiation step of said method comprises the steps of:

delivering said data packet, said data packet comprising a frame, to said control processor;

directing a request for said ARP lookup from said control processor to said master processor; and

searching a global ARP cache for a match with said destination address, said ARP cache comprising a table maintained by said master processor of ARP requests initiated within said routing apparatus, said routing apparatus comprising a switch.

76. The routing apparatus of Claim 75, said ARP lookup initiation step of said method further comprising one of the steps of:

if a match is found, returning the results of said cache search to said control processor;

if no match is found, issuing an ARP request by said master processor to ports connected to said subnet, receiving results of said request, and returning said results to said requesting control processor.

77. The routing apparatus of Claim 76, said ARP lookup initiation step of said method further comprising the step of:

when a subsequent frame is received at said switch having an identical destination address to a first frame for which an ARP request is pending, creating a table record for said destination address in said ARP cache indicating that a request for said address is pending, so that said control processor may receive said subsequent frame without directing multiple requests to said master processor for the same destination address.

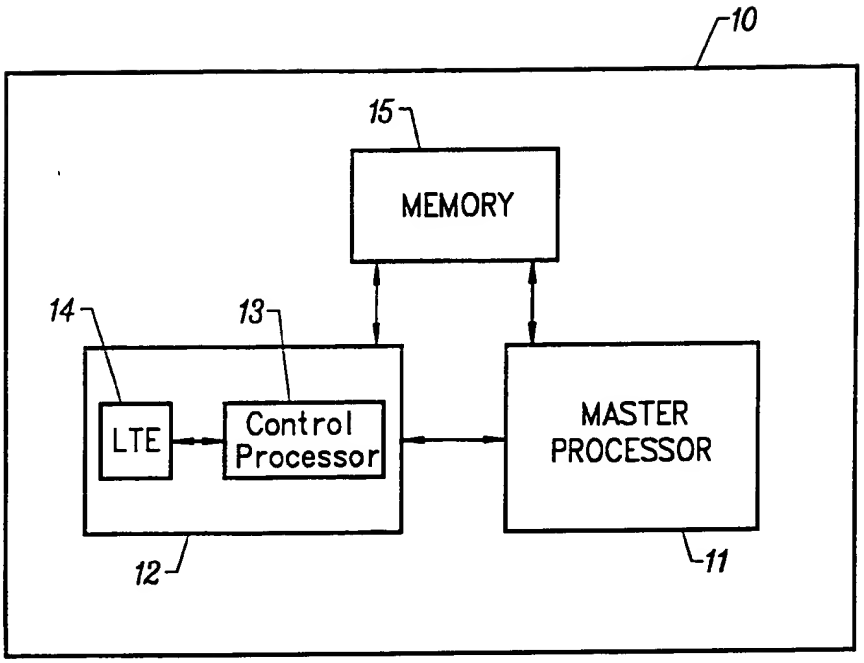
10
15
MEMORY
14
13
LTE
Control Processor
MASTER PROCESSOR
12
11

FIG. 1

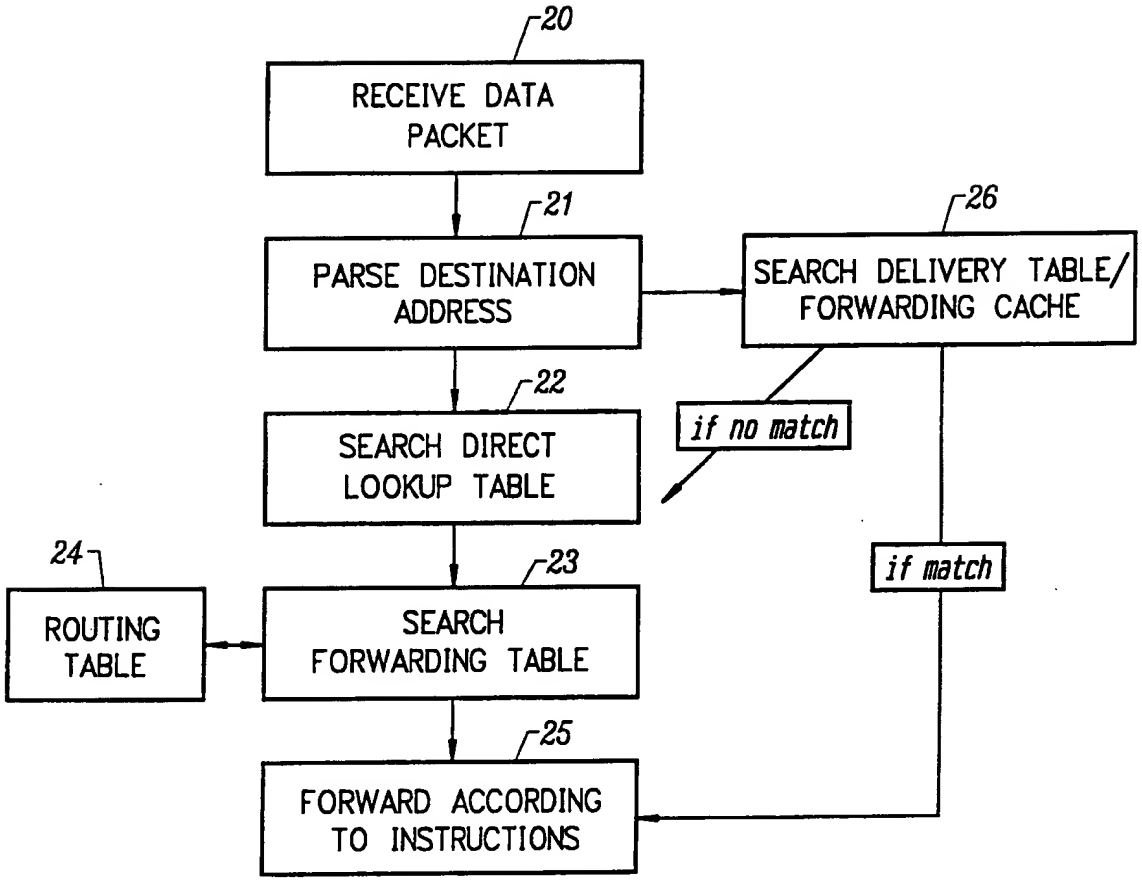
20
RECEIVE DATA PACKET
21
PARSE DESTINATION ADDRESS
26
SEARCH DELIVERY TABLE/ FORWARDING CACHE
if no match
22
SEARCH DIRECT LOOKUP TABLE
if match
24
ROUTING TABLE
23
SEARCH FORWARDING TABLE
25
FORWARD ACCORDING TO INSTRUCTIONS

FIG. 2